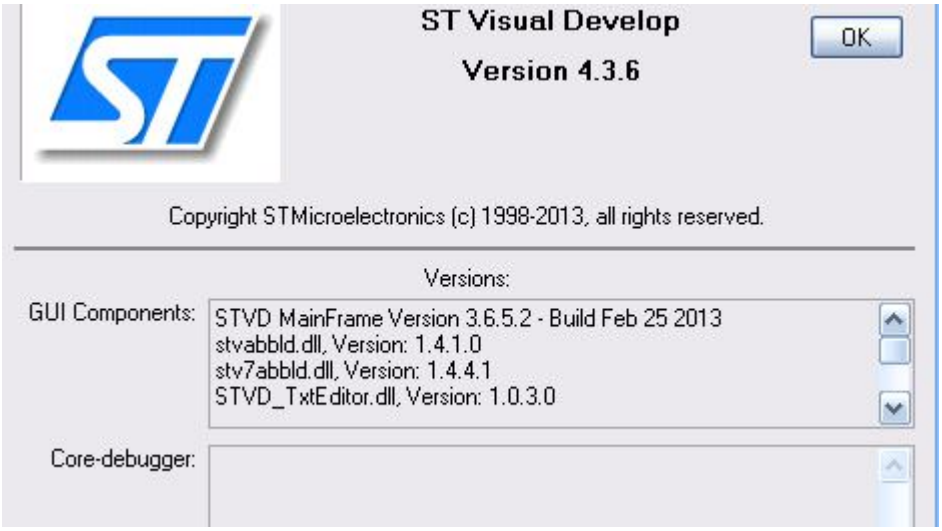
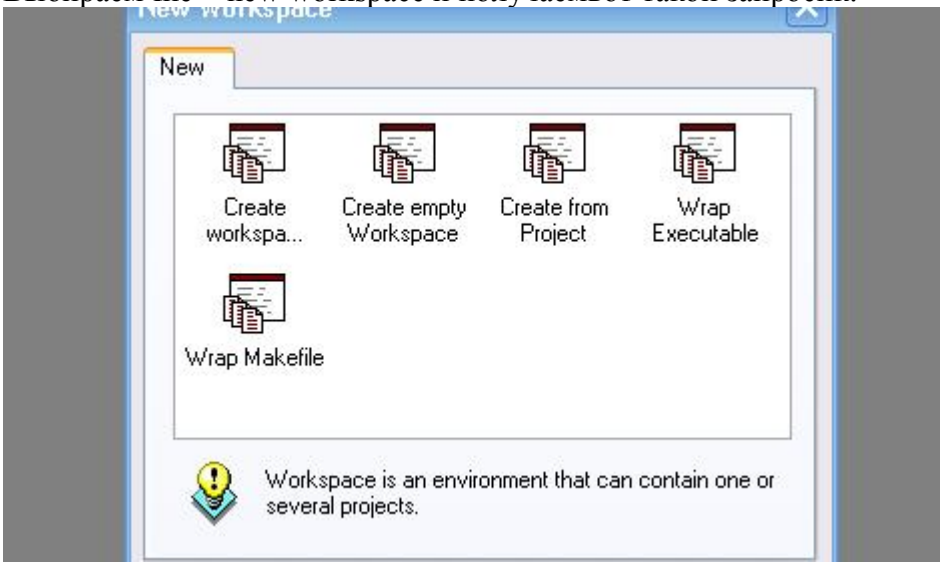
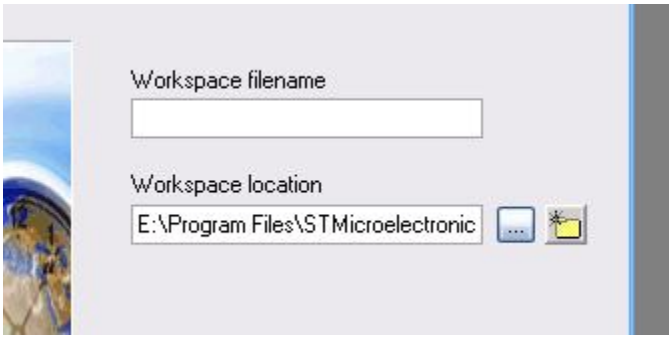
STM8
Пошаговая сборка проекта на ассемблере в среде IDE STVD v 4.3.6



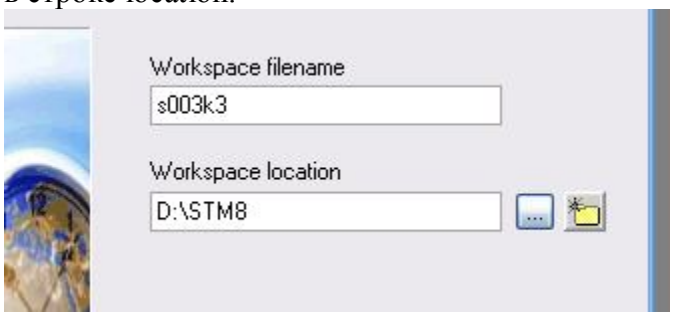
Сама IDE берется с сайта http://www.st.com
При установке обратить особое внимание на то, что у ПК имеющего интегрированный LPT порт инсталлятор запросит разрешение на установку драйвера для LPT порта, которую необходимо в обязательном порядке ЗАПРЕТИТЬ (во избежании проблем с восстановлением системы на ПК).
Предпочтительно задать папку для размещения проекта (в моем случае это D:\stm8).
Далее запускаем IDE.
Выбираем file – new workspace и получаемвот такой запросик:



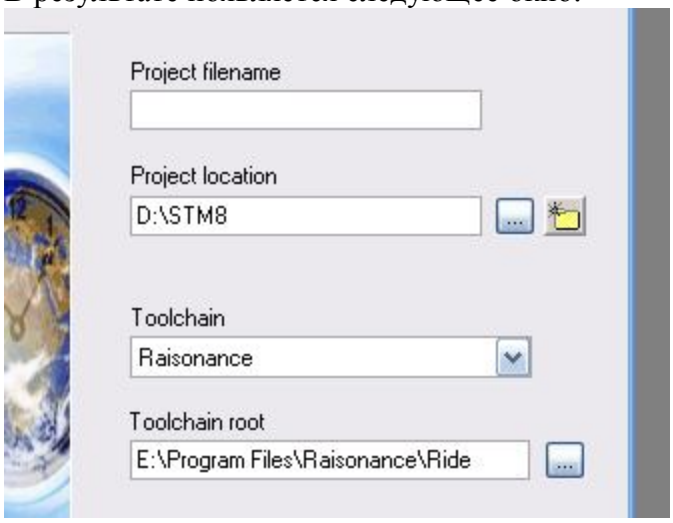
Из всего предлагаемого тыкаем create workspace и “ok”.
В выпавшем



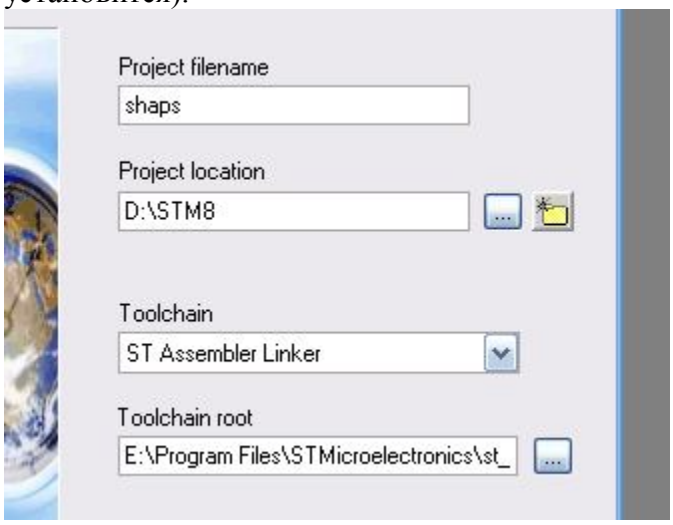
Набираю имя проекта S003k3 (или чего в голову взбредет) в окне filename и папку проекта в строке location:



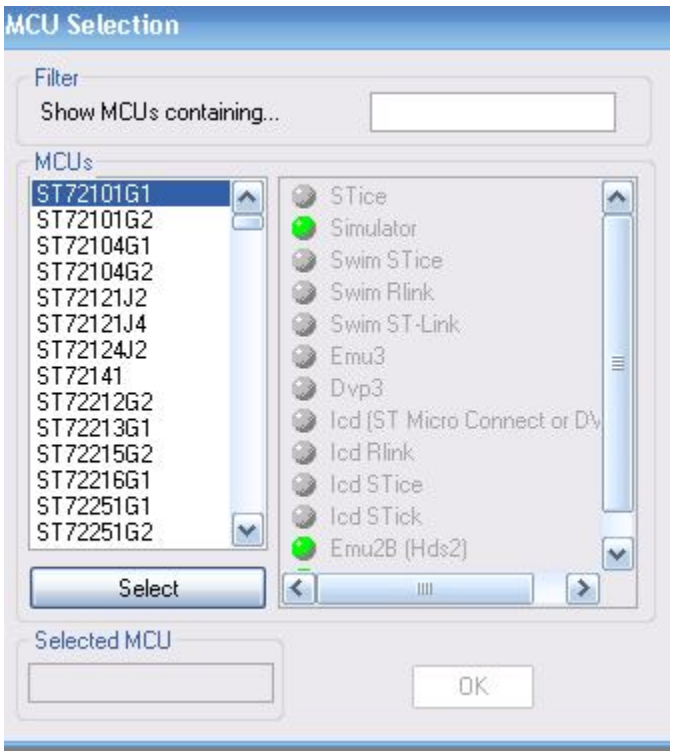
После чего тыцкаю «ок».
В результате появляется следующее окно:



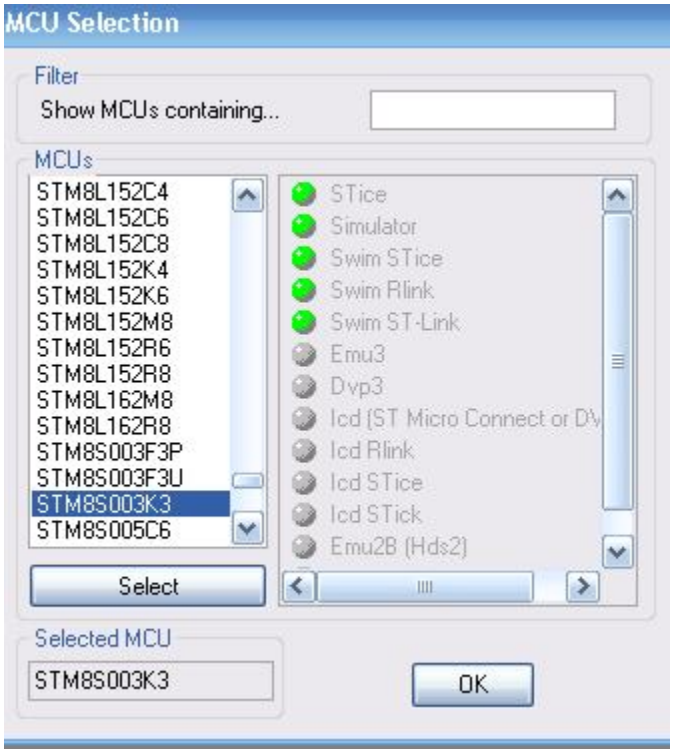
Здесь указываю конкретное имя проекта (коих в одной папке похоже может быть несколько) – на данный момент shaps, строку с location оставляю неизменной.
В строке toolchain выставляю ST assembler linker, toolchain root не трогаем (сама установится).



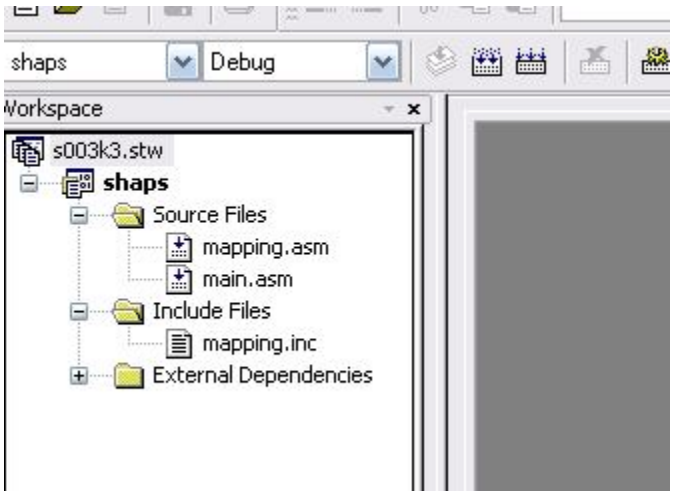
и тыцькаю «ок»…
В выпавшем окошке



выбираю stm8s003k3



И жму “select”…
В колонке workspace появился набор автоматически сгенерированных самой IDE начальных файлов:



Откроем те файлы во встроенном редакторе.

Сразу следует отметить, что встроенный редактор использует страницу win1251 – такую же надо указывать и во внешнем редакторе, ежли таковой будет использоваться, во избежании «кракозябр» при кириллице в комментариях.

Всего сгенерировано три файла:

Mapping.asm

Mapping.inc

и

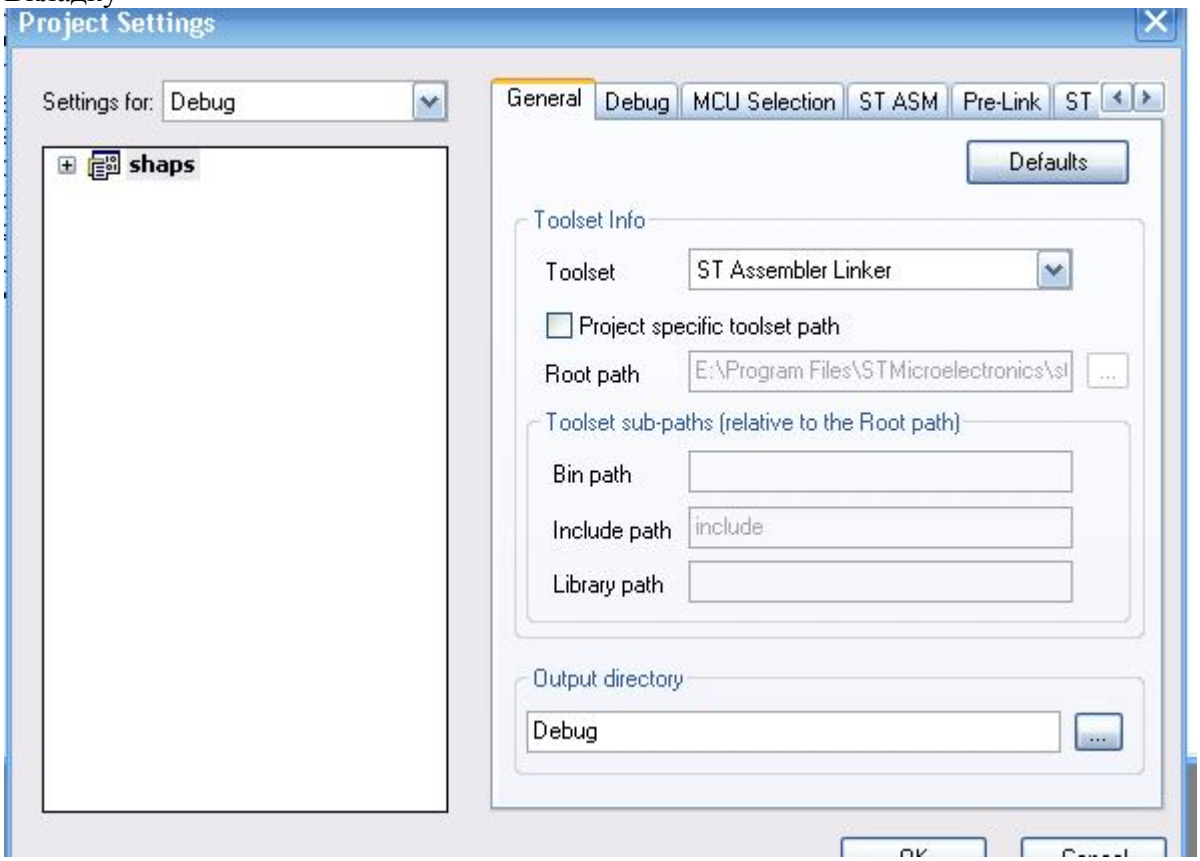
Main.asm

Два из них являются файлами автоматической разметки памяти с грифом «изменению не подлежат» - их так и оставляем без изменений.

А файл main используем как основу нашей «шапки»…

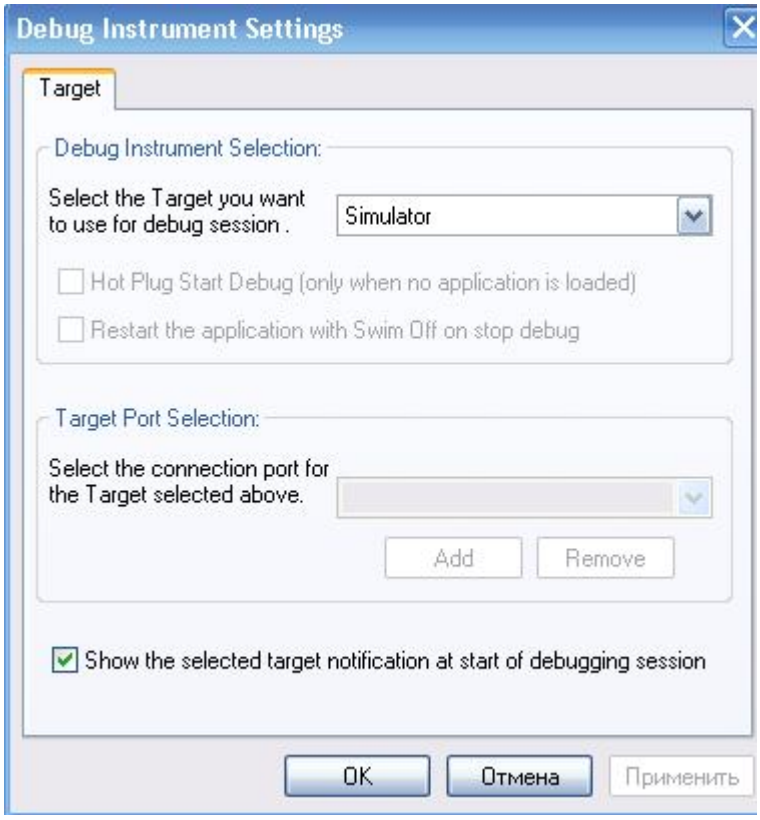
Теперь несколько слов по дополнительным «опционам»…

Вкладку



Пока трогать не рекомендуется – только «cansel» - она для «тонких настроек» опционами командной строки и прочими мелочами для весьма подготовленных пользователей.

В окошке



выбираю «симулятор» и жму последовательно «подтвердить» и «ок».
Несколько особенностей оформления текстов исходников:
Текст любого файла должен завершаться кодом (нажатием клавиши) «ВК» - т.е. в конце остается пустая строка. Если такого не выполнить при компиляции не будет учитываться последняя строка исходника (особо актуально для вложенных файлов).
Текст файла с расширением *.asm должен всегда начинаться строкой
Stm8/
- это указание на использование спецфайла (вероятно дополнительная инфотаблица или еще чего) C:\Program Files\STMicroelectronics\st_toolset\asm\stm8.tab .
Далее жмем «save workspace» и сохраняем полученный «ШКЕЛЕТ».

Заготовка шаблона собственного проекта.
В текущем окне редактора среды открываю новый файл и копирую в него самодельную шапку из проектов по MCS51 (предварительно открыв заготовку от MCS51 в DPAD).
Соответственно меняю оформление и строки под работу в качестве шаблон/заголовка для соответствия компилятору STM8.
Файлы вложений, в проектах для 51-й, АВР и ПИКов как *.txt у STM8 должны иметь расширение *.s .
Полученный суррогат оболочечного файла сохраняю как shell.asm в корневой папке проекта.
Открываю исходный main.asm, затем в окне workspace удаляю из проекта main.asm (он так только из перечня проекта удаляется, но не из папки!) и добавляю в раздел sourse files проекта свой шаблон.
Теперь необходимо отредактировать обязательное базовое содержимое в шаблоне оболочки.
Вверху исходного main.asm находятся сторчки:
stm8/

```
        #include "mapping.inc"
```

Их надо вписать в соответствующие места оболочки shell.asm. Кусок текста, касающегося векторов прерываний превращается в два файла
def_irq_tbl.s – собственно таблица векторов
и
irq_nnn.s – модуль «тел» обработчиков прерываний
Такой «изголяж» обусловлен особенностями записи как самих векторов, так и особенностями объявления подпрограмм выполнения прерываний.
Создаем def_irq_tbl.s и irq_nnn.s из покромсанного main.asm и также подключаем их в список проекта и в перечень #include в shell.asm.
Размещение самих #include в тексте shell.asm строго увязано в необходимой последовательности.
К сожалению, файл объявления имен регистров РСФ в исходном материале IDE оказался пустым…
Пришлось шкрябать самому, возводя матюки составителям даташита «от Сусанина» - в результате в проект вошли
sfr_STM8S003K3.inc – файл декларации имен для оболочки (и не только) и собственно
sfr_STM8S003K3.asm – файл объявления имен регистров РСФ (как public).
Для определения собственных сегментов ОЗУ/ПЗУ/EEPROM имен рабочих регистров и констант выделен файл my_prg_def.s .
Для определения собственных макросов файл my_prg_mac.s .
Файл-заготовки hard_init.s и soft_init.s собственно для начальной инициализации железа и программных модулей.
И собственно файлик для всяческих проб
my_tests.s
Перетаскиваю остатки из main.asm в soft_init.s , а стоп- заглушку
infinite_loop.l
```
        jra infinite_loop
```
в конец my_tests.s .
И создаю «пустышку» под макросы my_prg_mac.s с последующим подключением в окне проекта.
«шкелет» готов.
Проверяю пригодность к употреблению запустив rebuild oll.
Матюки…
Оказывается… директивы ассемблера должны (в отличии от меток) иметь отступ от начала строки минимум в пробел! А при копировании иногда автоматом придвигаются к началу строк (автонормирование встроенное в редактор IDE).

Теперь имея «ШКЕЛЕТ» можно приступать к экспериментам с системой команд.